



(19) Europäisches Patentamt  
European Patent Office  
Office européen des brevets



(11) EP 0 818 742 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:  
14.01.1998 Bulletin 1998/03

(51) Int Cl. 6: G06F 17/30, G06F 17/22

(21) Application number: 97304742.6

(22) Date of filing: 01.07.1997

(84) Designated Contracting States:  
AT BE CH DE DK ES FI FR GB GR IE IT LI LU MC  
NL PT SE

(72) Inventor: Renshaw, David Seager  
Winchester, Hampshire, SO22 6NN (GB)

(30) Priority: 11.07.1996 GB 9614570

(74) Representative: Davies, Simon Robert  
IBM  
UK Intellectual Property Department  
Hursley Park  
Winchester, Hampshire SO21 2JN (GB)

(71) Applicant: International Business Machines Corporation  
Armonk, N.Y. 10504 (US)

(54) Embedded HTML documents

(57) The present invention relates to embedded HTML documents and to a method and system for rendering such documents to a visual display unit. The Internet does not currently support embedded documents. The current manner of viewing a plurality of HTML documents is to include a URL within one documents which allows access to a second document. HTML does not provide for the nesting or embedding of

HTML documents. Accordingly the present invention provides a system and method for realising embedded HTML documents. The present invention utilises a Java applet which can be launched by a Java enabled browser. The Java applet can parse and render HTML instructions contained within an HTML document to a reserved area of a visual display unit. The Java applet can also launch further applets and therefore allow further nesting or embedding of HTML documents.

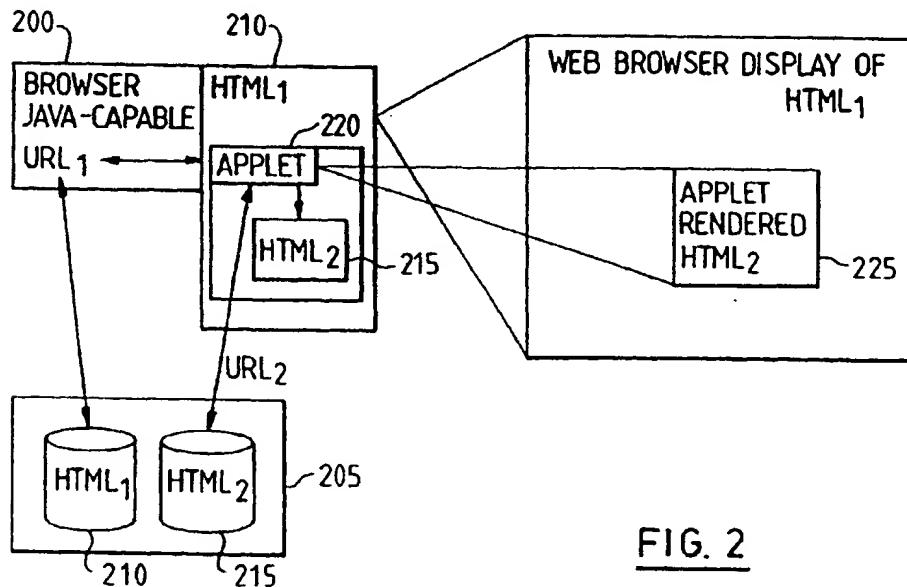


FIG. 2

**Description**

The present invention relates to embedded HTML documents and to a system and method for the rendering thereof.

The use and importance of the Internet as a means of exchanging information over recent years has increased dramatically. Various businesses are now using the Internet to advertise their products and services. The operation of the Internet and the use of html is well understood within the art and therefore only brief details thereof will be presented here. Further information relating to the Internet and HTML can be found in, for example, "HTML Sourcebook, A complete guide to HTML", by Graham, I. S. (John Wiley and Sons, Inc), "Spinning the web" by Andrew Ford (International Thomson Publishing, London 1995) and "The World Wide web Unleashed" by John December and Neil Randall (SAMS Publishing, Indianapolis 1994). Information accessed via the Internet is typically displayed using a web Browser. The web Browser can interpret the HTML contained within an HTML document or web page. The HTML controls the presentation of output to a visual display unit of the computer running the web Browser as is well known within the art. A web page typically contains so-called hyperlinks to other web pages. Each hyperlink is a Uniform Resource Locator (URL) which is used to access further a web page or HTML document stored on a particular server connected to the Internet. In order to display the other web pages, the user selects the URL corresponding to that web page using a mouse. The associated web page or HTML document is retrieved and forwarded to the user's web Browser for subsequent display in the conventional manner. Selecting a URL from one web page causes the web page identified by that URL to be displayed in place of the currently displayed web page thereby removing the currently viewed web page from view.

A compound document is a document which comprises many component parts. Each component part typically contains and displays data of a particular type. For example, the text of a patent application may be contained within a text component of a document while the accompanying drawings are contained within graphics component of the document. The text and graphics components can manipulate their respective data independently of each other. Furthermore, the underlying code controlling the display or processing of the data associated therewith can be modified or replaced independently of the other component parts constituting the document. Many environments exist for the development of compound documents. An example of such an environment is the OpenDoc environment available from Component Integration Laboratories (Sunnyvale, California, USA), and described in "The Byte Guide to OpenDoc" by Andrew McBride and Joshua Susser, Osborne/McGraw-Hill, Berkeley Ca, 1996, ISBN 0-07-882118-5. Further information relating to the struc-

ture of compound documents can be found in the above reference.

The HTML documents or web pages as described above can be utilised to produce single layer documents which comprise different data types. For example, the <IMG SRC="filename.gif"> tag can be used to include an image within a rendered HTML document. The image contained in the file "filename.gif" is included by the web browser in the rendered HTML page. The current documents accessible on the Internet can only be viewed as single layer documents comprising text and images. The images within a rendered HTML documents are rendered by the web Browser. HTML does not support nested or multi-layered HTML documents.

Accordingly, the present invention provides a method for rendering first and second html documents comprising respective first and second sets of html data to a display screen, wherein the first html document has embedded therein the second html document, comprising the steps of: parsing and rendering the first set of html data to a first area of said screen, reserving a second area of said screen for use in rendering said second set of html data, said second area being embedded within said first area, and parsing and rendering the second set of html data to said screen within said second area.

The present invention advantageously provides multi-layer or embedded HTML documents and allows such documents to be rendered to the screen of a computer system or other suitable display device.

In the preferred embodiment, said first html document includes a Java applet tag containing a reference to a Java html parsing and rendering applet for parsing and rendering htm1 to the screen, and the step of reserving the second area is responsive to said Java applet tag, and said steps of parsing and rendering said second set of html data are performed by said Java applet.

This embodiment enables embedded HTML documents to be realised and rendered independently of the nature of the underlying computer system providing that the latter has the capability of supporting Java. Such support could be provided in many instances by way of a Java-capable Web Browser such as the HotJava browser available from Sun MicroSystems. Of course, any other network-enabled language could be used instead of Java to provide the parsing and rendering function.

The parsing and rendering of embedded HTML documents may employ the same text format and graphics fonts within both HTML documents, or alternatively they may differ between nested HTML documents. Thus it is preferred that the method further comprises the steps of creating a data structure for storing format data comprising a plurality of characteristics which determine the format of the rendering, said rendering being performed according to current format data, and wherein said steps of parsing and rendering further comprise the steps of: identifying new format data, storing a copy of said cur-

rent format data in the data structure, and modifying the current format data according to the new format data.

In the preferred embodiment, when a need to change the current format data to a preceding format is identified, a copy of the preceding format data stored in the data structure may be recovered, and the current format data modified according to the recovered format data.

Preferably the step of modifying comprises the step of changing only selectable ones of said plurality of characteristics of the current format data according to the new format data, thereby retaining selectable characteristics of stored format data. Utilising this approach enables a reduction in processing to be realised. Rather than having to set new format parameters each time a format changes, the previous parameters can be used. For example, if the text merely changes from a plain style to an italicised style, it is desirable that the font size and name remains the same. Therefore, the current characteristics would retain the current font size and name and vary only the font style.

The invention also provides a system for rendering first and second html documents comprising respective first and second sets of html data to a display screen, wherein the first html document has embedded therein the second html document, comprising means for parsing and rendering the first set of html data to a first area of said screen, means for reserving a second area of said screen for use in rendering said second set of html data, said second area being embedded within said first area, and means for parsing and rendering the second set of html data to said screen within said second area.

Embodiments of the present invention will now be described, by way of example only, with reference to the accompanying drawings in which:

figure 1 illustrates a computer system;

figure 2 illustrates schematically the operation of the present invention;

figure 3 depicts schematically a rendered multi-level or embedded HTML document;

figure 4 shows a schematic flow diagram of the Java applet parser and renderer; and

figure 5 shows a table for mapping HTML tags and corresponding formats.

Referring to figure 1, there is depicted a block diagram of a personal computer system 10, such as an IBM PS/2 personal computer. The computer system 10 includes a 32-bit processor and system bus 12. Connected to system bus 12 is a central processing unit (CPU) 14, for example an Intel Pentium Processor or equivalent microprocessor. CPU 14 passes data to and receives data from other devices attached to system bus

12 over the system bus. Traffic on the bus is controlled by a bus controller 16. An interrupt controller 18 handles interrupts passed between CPU 14 and the remaining devices.

5 Read only memory (ROM) 20 is non-volatile memory storing power on test processes and a basic input/output system (BIOS or ROM-BIOS). The system memory 22 is random access memory into which an operating system, preferably a 32-bit operating system such as IBM OS/2 WARP which supports concurrent processes or multiple concurrent threads, is loaded for execution to support execution of applications. The computer system has software or applications which provide for the execution of Java. Additional hardware components of computer 10 attached to system bus 12 include a memory controller 24 and a system configuration store

10 26, provided by random access memory (RAM). 15 26, provided by random access memory (RAM). 20 Auxiliary data storage is provided by peripheral controllers and associated storage devices including, a floppy disk or diskette controller 28 and drive 30, a hard drive controller 32 and hard drive device 34, and a compact disk (CD) read only memory (ROM) controller 36 with CD-ROM drive 38. An SVGA controller 40 includes a RAM buffer 44 in which the current frame for display

25 42 is stored. In some computers, the buffer 44 may be loaded directly from system memory 22 or from an auxiliary storage controller. 30 Direct memory access (DMA) controller 46 handles data transfers between auxiliary storage devices or other input/output devices, and system memory 22 without interaction by CPU 14. Keyboard controller 48 provides an interface to a keyboard 50, for user entries, and may be used to provide an interface to a "mouse". Parallel controller 52 is a device controller for an input or output device (e.g. a printer connected to computer 10 by a parallel cable). A camera 56 may also be provided for allowing the computer to received video signal via video input 54.

35 It will be appreciated that many variations are 40 known to the skilled person regarding the workstation of figure 1. One possibility, for example, is for the workstation of figure 1 to be replaced by a so-called "network computer", which has no hard disk drive of its own, but rather downloads instructions across a network. 45 Referring now to figure 2, there is shown a web Browser 200 having Java capability, a so-called Java-capable browser, used to access html documents or web pages stored for example, on an Internet server 205 (alternatively the html documents may be stored on a 50 server accessible via any other suitable network, such as an intranet, local area network, etc). The Java-capable web browser may be, for example, the HotJava web browser available from Sun MicroSystems. The Internet server 205 contains at least two HTML documents 210 and 215, both containing HTML instructions to be rendered to the screen of the computer system running the Java-capable browser. The first HTML document 210 contains a plurality of HTML instructions and includes

an applet tag which causes a Java applet 220 to be downloaded and executed. The Java applet 220 is an HTML parser capable of parsing and rendering HTML instructions to the screen of the computer system upon which the browser is running. The creation of applets and their incorporation into HTML documents or web pages is well known within the art. Further information relating to applets can be found in, for example, "Teach Yourself Java in 21 Days" by L. Lemay and C. L. Perkins (published by SAMSNET, 201 West 103rd Street, Indianapolis, Indiana, 46290). Typically, the applet tag has associated therewith a "param" tag which contains a URL of data to be rendered by the applet. The applet tag also contains "width" and "height" parameters which govern the size of the work space 225 within which an applet can draw or render data. In the present embodiment, the URL points to the second HTML document 215 to be rendered to the screen 42 of the computer system. When the applet is launched, the second HTML document 215 is obtained from the Internet server and the HTML instructions contained therein are rendered to the area of the screen reserved for use by the applet. The rendering by the applet within the respective reserved area of the screen and the actions performed by the applet are independent of the actions performed by the HTML instructions contained within the first HTML document 210. Therefore, using the Java HTML parsing and rendering applet, a HTML document can be embedded and rendered within another HTML document thereby allowing the creation of multi-layered or embedded HTML documents having at least two layers of HTML.

In a further embodiment of the present invention, the HTML parsing and rendering applet includes the capability to parse applet tags and associated tags and to launch applets accordingly. Therefore, further HTML documents can be embedded within the second HTML document by including therein an HTML parsing and rendering applet and a reference to yet another HTML file thereby adding a further level to the multi-level or embedded HTML document. As a consequence of nesting HTML parsers and renderers according to the above embodiment, multi-level or embedded HTML documents may be created and rendered comprising more than two levels of HTML. Each succeeding or embedded HTML document is realised by including an HTML parser and rendering applet within a preceding HTML document.

The following HTML document illustrates a multi-levelled HTML document realised according to an embodiment of the present invention.

```

<html>
<head>
<title>EMBEDDED DOCUMENT EXAMPLE</title>
</head>
<body>
<applet codebase=http://bull.hursley.ibm.com/

```

5 classes
code=HTMLpart.class width=600 height=400>
<param name=dataURL
value=http://bull.hursley.ibm.com/example1a.
html>
</applet>
</body>

#### EXAMPLE 1

10 The HTML elements such as <html>, <head>, <title>, <body> and </body> are well understood by those familiar with HTML and will not be commented upon further. Within the applet tag, <applet...>, "code" is used to indicate the name of the class file that holds the applet which parses and renders HTML and "codebase" is used to identify the Internet server and directory which contains the class file containing the applet. Therefore, in the present example, the HTML parser applet would 15 be located in the class called "HTMLpart.class" on Internet server "bull.hursley.ibm.com" in a directory called "classes". The "param" tag is used to pass parameters to applets as is well known within the art. Therefore an applet can receive inputs from the HTML document 20 which contains the <applet> tag. The <param> tag comprises two elements, namely a name and a value which represent the name of the parameter to be passed to the applet and the value of the parameter passed to the applet respectively. It can be seen from the above example, that the parameter name is "dataURL" and the parameter value is "http://bull.hursley.ibm.com/example1a.html". The parameters are incorporated into the applet using the "getParameter" method as is well known within the art.

25 When the applet is executed the "HTMLpart" obtains the "dataURL" parameter using intrinsic Java capabilities and thereby obtains the contents of the file "example1a.html" from Internet server "bull.hursley.ibm.com". The file "example1a.html" is a file containing further HTML instructions. The file "example1a.html" may 30 contain, for example, the following HTML instructions:

35
40
45
50
55
60
65
70
75
80
85
90
95
100
105
110
115
120
125
130
135
140
145
150
155
160
165
170
175
180
185
190
195
200
205
210
215
220
225
230
235
240
245
250
255
260
265
270
275
280
285
290
295
300
305
310
315
320
325
330
335
340
345
350
355
360
365
370
375
380
385
390
395
400
405
410
415
420
425
430
435
440
445
450
455
460
465
470
475
480
485
490
495
500
505
510
515
520
525
530
535
540
545
550
555
560
565
570
575
580
585
590
595
600
605
610
615
620
625
630
635
640
645
650
655
660
665
670
675
680
685
690
695
700
705
710
715
720
725
730
735
740
745
750
755
760
765
770
775
780
785
790
795
800
805
810
815
820
825
830
835
840
845
850
855
860
865
870
875
880
885
890
895
900
905
910
915
920
925
930
935
940
945
950
955
960
965
970
975
980
985
990
995
1000
1005
1010
1015
1020
1025
1030
1035
1040
1045
1050
1055
1060
1065
1070
1075
1080
1085
1090
1095
1100
1105
1110
1115
1120
1125
1130
1135
1140
1145
1150
1155
1160
1165
1170
1175
1180
1185
1190
1195
1200
1205
1210
1215
1220
1225
1230
1235
1240
1245
1250
1255
1260
1265
1270
1275
1280
1285
1290
1295
1300
1305
1310
1315
1320
1325
1330
1335
1340
1345
1350
1355
1360
1365
1370
1375
1380
1385
1390
1395
1400
1405
1410
1415
1420
1425
1430
1435
1440
1445
1450
1455
1460
1465
1470
1475
1480
1485
1490
1495
1500
1505
1510
1515
1520
1525
1530
1535
1540
1545
1550
1555
1560
1565
1570
1575
1580
1585
1590
1595
1600
1605
1610
1615
1620
1625
1630
1635
1640
1645
1650
1655
1660
1665
1670
1675
1680
1685
1690
1695
1700
1705
1710
1715
1720
1725
1730
1735
1740
1745
1750
1755
1760
1765
1770
1775
1780
1785
1790
1795
1800
1805
1810
1815
1820
1825
1830
1835
1840
1845
1850
1855
1860
1865
1870
1875
1880
1885
1890
1895
1900
1905
1910
1915
1920
1925
1930
1935
1940
1945
1950
1955
1960
1965
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
2020
2025
2030
2035
2040
2045
2050
2055
2060
2065
2070
2075
2080
2085
2090
2095
2100
2105
2110
2115
2120
2125
2130
2135
2140
2145
2150
2155
2160
2165
2170
2175
2180
2185
2190
2195
2200
2205
2210
2215
2220
2225
2230
2235
2240
2245
2250
2255
2260
2265
2270
2275
2280
2285
2290
2295
2300
2305
2310
2315
2320
2325
2330
2335
2340
2345
2350
2355
2360
2365
2370
2375
2380
2385
2390
2395
2400
2405
2410
2415
2420
2425
2430
2435
2440
2445
2450
2455
2460
2465
2470
2475
2480
2485
2490
2495
2500
2505
2510
2515
2520
2525
2530
2535
2540
2545
2550
2555
2560
2565
2570
2575
2580
2585
2590
2595
2600
2605
2610
2615
2620
2625
2630
2635
2640
2645
2650
2655
2660
2665
2670
2675
2680
2685
2690
2695
2700
2705
2710
2715
2720
2725
2730
2735
2740
2745
2750
2755
2760
2765
2770
2775
2780
2785
2790
2795
2800
2805
2810
2815
2820
2825
2830
2835
2840
2845
2850
2855
2860
2865
2870
2875
2880
2885
2890
2895
2900
2905
2910
2915
2920
2925
2930
2935
2940
2945
2950
2955
2960
2965
2970
2975
2980
2985
2990
2995
3000
3005
3010
3015
3020
3025
3030
3035
3040
3045
3050
3055
3060
3065
3070
3075
3080
3085
3090
3095
3100
3105
3110
3115
3120
3125
3130
3135
3140
3145
3150
3155
3160
3165
3170
3175
3180
3185
3190
3195
3200
3205
3210
3215
3220
3225
3230
3235
3240
3245
3250
3255
3260
3265
3270
3275
3280
3285
3290
3295
3300
3305
3310
3315
3320
3325
3330
3335
3340
3345
3350
3355
3360
3365
3370
3375
3380
3385
3390
3395
3400
3405
3410
3415
3420
3425
3430
3435
3440
3445
3450
3455
3460
3465
3470
3475
3480
3485
3490
3495
3500
3505
3510
3515
3520
3525
3530
3535
3540
3545
3550
3555
3560
3565
3570
3575
3580
3585
3590
3595
3600
3605
3610
3615
3620
3625
3630
3635
3640
3645
3650
3655
3660
3665
3670
3675
3680
3685
3690
3695
3700
3705
3710
3715
3720
3725
3730
3735
3740
3745
3750
3755
3760
3765
3770
3775
3780
3785
3790
3795
3800
3805
3810
3815
3820
3825
3830
3835
3840
3845
3850
3855
3860
3865
3870
3875
3880
3885
3890
3895
3900
3905
3910
3915
3920
3925
3930
3935
3940
3945
3950
3955
3960
3965
3970
3975
3980
3985
3990
3995
4000
4005
4010
4015
4020
4025
4030
4035
4040
4045
4050
4055
4060
4065
4070
4075
4080
4085
4090
4095
4100
4105
4110
4115
4120
4125
4130
4135
4140
4145
4150
4155
4160
4165
4170
4175
4180
4185
4190
4195
4200
4205
4210
4215
4220
4225
4230
4235
4240
4245
4250
4255
4260
4265
4270
4275
4280
4285
4290
4295
4300
4305
4310
4315
4320
4325
4330
4335
4340
4345
4350
4355
4360
4365
4370
4375
4380
4385
4390
4395
4400
4405
4410
4415
4420
4425
4430
4435
4440
4445
4450
4455
4460
4465
4470
4475
4480
4485
4490
4495
4500
4505
4510
4515
4520
4525
4530
4535
4540
4545
4550
4555
4560
4565
4570
4575
4580
4585
4590
4595
4600
4605
4610
4615
4620
4625
4630
4635
4640
4645
4650
4655
4660
4665
4670
4675
4680
4685
4690
4695
4700
4705
4710
4715
4720
4725
4730
4735
4740
4745
4750
4755
4760
4765
4770
4775
4780
4785
4790
4795
4800
4805
4810
4815
4820
4825
4830
4835
4840
4845
4850
4855
4860
4865
4870
4875
4880
4885
4890
4895
4900
4905
4910
4915
4920
4925
4930
4935
4940
4945
4950
4955
4960
4965
4970
4975
4980
4985
4990
4995
5000
5005
5010
5015
5020
5025
5030
5035
5040
5045
5050
5055
5060
5065
5070
5075
5080
5085
5090
5095
5100
5105
5110
5115
5120
5125
5130
5135
5140
5145
5150
5155
5160
5165
5170
5175
5180
5185
5190
5195
5200
5205
5210
5215
5220
5225
5230
5235
5240
5245
5250
5255
5260
5265
5270
5275
5280
5285
5290
5295
5300
5305
5310
5315
5320
5325
5330
5335
5340
5345
5350
5355
5360
5365
5370
5375
5380
5385
5390
5395
5400
5405
5410
5415
5420
5425
5430
5435
5440
5445
5450
5455
5460
5465
5470
5475
5480
5485
5490
5495
5500
5505
5510
5515
5520
5525
5530
5535
5540
5545
5550
5555
5560
5565
5570
5575
5580
5585
5590
5595
5600
5605
5610
5615
5620
5625
5630
5635
5640
5645
5650
5655
5660
5665
5670
5675
5680
5685
5690
5695
5700
5705
5710
5715
5720
5725
5730
5735
5740
5745
5750
5755
5760
5765
5770
5775
5780
5785
5790
5795
5800
5805
5810
5815
5820
5825
5830
5835
5840
5845
5850
5855
5860
5865
5870
5875
5880
5885
5890
5895
5900
5905
5910
5915
5920
5925
5930
5935
5940
5945
5950
5955
5960
5965
5970
5975
5980
5985
5990
5995
6000
6005
6010
6015
6020
6025
6030
6035
6040
6045
6050
6055
6060
6065
6070
6075
6080
6085
6090
6095
6100
6105
6110
6115
6120
6125
6130
6135
6140
6145
6150
6155
6160
6165
6170
6175
6180
6185
6190
6195
6200
6205
6210
6215
6220
6225
6230
6235
6240
6245
6250
6255
6260
6265
6270
6275
6280
6285
6290
6295
6300
6305
6310
6315
6320
6325
6330
6335
6340
6345
6350
6355
6360
6365
6370
6375
6380
6385
6390
6395
6400
6405
6410
6415
6420
6425
6430
6435
6440
6445
6450
6455
6460
6465
6470
6475
6480
6485
6490
6495
6500
6505
6510
6515
6520
6525
6530
6535
6540
6545
6550
6555
6560
6565
6570
6575
6580
6585
6590
6595
6600
6605
6610
6615
6620
6625
6630
6635
6640
6645
6650
6655
6660
6665
6670
6675
6680
6685
6690
6695
6700
6705
6710
6715
6720
6725
6730
6735
6740
6745
6750
6755
6760
6765
6770
6775
6780
6785
6790
6795
6800
6805
6810
6815
6820
6825
6830
6835
6840
6845
6850
6855
6860
6865
6870
6875
6880
6885
6890
6895
6900
6905
6910
6915
6920
6925
6930
6935
6940
6945
6950
6955
6960
6965
6970
6975
6980
6985
6990
6995
7000
7005
7010
7015
7020
7025
7030
7035
7040
7045
7050
7055
7060
7065
7070
7075
7080
7085
7090
7095
7100
7105
7110
7115
7120
7125
7130
7135
7140
7145
7150
7155
7160
7165
7170
7175
7180
7185
7190
7195
7200
7205
7210
7215
7220
7225
7230
7235
7240
7245
7250
7255
7260
7265
7270
7275
7280
7285
7290
7295
7300
7305
7310
7315
7320
7325
7330
7335
7340
7345
7350
7355
7360
7365
7370
7375
7380
7385
7390
7395
7400
7405
7410
7415
7420
7425
7430
7435
7440
7445
7450
7455
7460
7465
7470
7475
7480
7485
7490
7495
7500
7505
7510
7515
7520
7525
7530
7535
7540
7545
7550
7555
7560
7565
7570
7575
7580
7585
7590
7595
7600
7605
7610
7615
7620
7625
7630
7635
7640
7645
7650
7655
7660
7665
7670
7675
7680
7685
7690
7695
7700
7705
7710
7715
7720
7725
7730
7735
7740
7745
7750
7755
7760
7765
7770
7775
7780
7785
7790
7795
7800
7805
7810
7815
7820
7825
7830
7835
7840
7845
7850
7855
7860
7865
7870
7875
7880
7885
7890
7895
7900
7905
7910
7915
7920
7925
7930
7935
7940
7945
7950
7955
7960
7965
7970
7975
7980
7985
7990
7995
8000
8005
8010
8015
8020
8025
8030
8035
8040
8045
8050
8055
8060
8065
8070
8075
8080
8085
8090
8095
8100
8105
8110
8115
8120
8125
8130
8135
8140
8145
8150
8155
8160
8165
8170
8175
8180
8185
8190
8195
8200
8205
8210
8215
8220
8225
8230
8235
8240
8245
8250
8255
8260
8265
8270
8275
8280
8285
8290
8295
8300
8305
8310
8315
8320
8325
8330
8335
8340
8345
8350
8355
8360
8365
8370
8375
8380
8385
8390
8395
8400
8405
8410
8415
8420
8425
8430
8435
8440
8445
8450
8455
8460
8465
8470
8475
8480
8485
8490
8495
8500
8505
8510
8515
8520
8525
8530
8535
8540
8545
8550
8555
8560
8565
8570
8575
8580
8585
8590
8595
8600
8605
8610
8615
8620
8625
8630
8635
8640
8645
8650
8655
8660
8665
8670
8675
8680
8685
8690
8695
8700
8705
8710
8715
8720
8725
8730
8735
8740
8745
8750
8755
8760
8765
8770
8775
8780
8785
8790
8795
8800
8805
8810
8815
8820
8825
8830
8835
8840
8845
8850
8855
8860
8865
8870
8875
8880
8885
8890
8895
8900
8905
8910
8915
8920
8925
8930
8935
8940
8945
8950
8955
8960
8965
8970
8975
8980
8985
8990
8995
9000
9005
9010
9015
9020
9025
9030
9035
9040
9045
9050
9055
9060
9065
9070
9075
9080
9085
9090
9095
9100
9105
9110
9115
9120
9125
9130
9135
9140
9145
9150
9155
9160
9165
9170
9175
9180
9185
9190
9195
9200
9205
9210
9215
9220
9225
9230
9235
9240
9245
9250
9255
9260
9265
9270
9275
9280
9285
9290
9295
9300
9305
9310
9315
9320
9325
9330
9335
9340
9345
9350
9355
9360
9365
9370
9375
9380
9385
9390
9395
9400
9405
9410
9415
9420
9425
9430
9435
9440
9445
9450
9455
9460
9465
9470
9475
9480
9485
9490
9495
9500
9505
9510
9515
9520
9525
9530
9535
9540
9545
9550
9555
9560
9565
9570
9575
9580
9585
9590
9595
9600
9605
9610
9615
9620
9625
9630
9635
9640
9645
9650
9655
9660
9665
9670
9675
9680
9685
9690
9695
9700
9705
9710
9715
9720
9725
9730
9735
9740
9745
9750
9755
9760
9765
9770
9775
9780
9785
9790
9795
9800
9805
9810
9815
9820
9825
9830
9835
9840
9845
9850
9855
9860
9865
9870
9875
9880
9885
9890
9895
9900
9905
9910
9915
9920
9925
9930
9935
9940
9945
9950
9955
9960
9965
9970
9975
9980
9985
9990
9995
9999
10000
10005
10010
10015
10020
10025
10030
10035
10040
10045
10050
10055
10060
10065
10070
10075
10080
10085
10090
10095
10100
10105
10110
10115
10120
10125
10130
10135
10140
10145
10150
10155
10160
10165
10170
10175
10180
10185
10

```
</applet>
</body>
```

#### EXAMPLE 2

The HTML file "example1b.html" referred to above is processed by the applet HTMLpart.class, which parses and renders the HTML instructions shown below. The HTML instructions below render text and an accompanying image to the screen 42 of the computer system.

```
<html>
<head>
<title>Embedded text and image</title>
</head>
<body>
```

This section of text is related to the image rendered below

```

</body>
```

#### EXAMPLE 3

The applet tag in EXAMPLE 2 reserves, within the work space reserved for the first applet tag in EXAMPLE 1, a further work space for rendering the output of the second applet. Therefore, the first HTML document above in EXAMPLE 1 has embedded therein the HTML document shown in EXAMPLE 2 which, in turn, has embedded therein the HTML document shown in EXAMPLE 3 thereby producing a multi-levelled or embedded HTML document. Each level of HTML document can process and render data independently of the other levels of HTML. Therefore, compound documents can be constructed by having different data types processed and rendered by respective HTML documents (although of course the different HTML components do not necessarily contain different data types).

Referring to figure 3 there is shown schematically the result of rendering the multi-levelled HTML document illustrated in Examples 1-3. The work space reserved for use by each applet has been depicted by a dotted line for the purpose of illustration only. Normally, the dotted lines would not be present, to allow seamless integration of one html document within another.

Referring to figures 4A to 4C, there is shown a schematic flow diagram of the processing performed by the Java applet parser and renderer. At step 400 the default text format options are set and a text buffer and a format stack are created. The text format options govern the manner in which the text is subsequently displayed upon the screen of the computer. In the present embodiment, the text format options include font type, font size and font style. The text format options are stored in a data object and can take various values according to the capabilities of the computer executing the Java applet. The

default options may be, for example, a font type of Times Roman, a font size of 10 and a font style of plain. The text buffer is used to store characters to be rendered to the screen of the computer system by the applet and the format stack is used to hold temporarily a plurality of text formats. The text data format object is used to store text format options on the text format stack.

5 The HTML document to be parsed and rendered by the applet is stored as a stream of contiguous characters. Hence the HTML instructions of, for example, EXAMPLE 3 would be stored as follows:

<html><head><title>Embedded text and image</title></head><body>This section of text is related to the image rendered below </body>.

10 15 The text buffer is emptied at step 405 in preparation for receiving characters to be rendered. The characters constituting the HTML are read from a given HTML file and stored in the text buffer at step 410 until a "<" character is encountered. The "<" character within HTML 20 represents the beginning of a HTML tag as is well known within the art. A determination is made at step 415 as to whether or not the text buffer is empty. If the text buffer is not empty, the text currently stored therein is rendered to the screen using the standard Java libraries and the 25 current format setting at step 420 and then the text buffer is cleared at step 425 in preparation for receiving further text to be rendered. If the text buffer is empty, further characters are read from the HTML document at step 430 until a ">" character is encountered.

30 35 A determination is made at step 435 as to whether or not the first character following the "<" is an exclamation character, "!". If so, the characters read from the HTML document since the last "<" character represent comments within the HTML document which are not intended to be rendered to the screen and control returns to step 405 whereupon the text buffer is cleared. If not, a determination is made at step 440 as to whether or not the next non-space character is a solidus thereby indicating that the tag represents an end tag.

40 45 50 55 If the next character is not a solidus, the characters stored within the text buffer must represent an HTML tag. A determination is made at step 445 as to the type of HTML tag represented by the characters read by the applet by comparing the characters stored in the text buffer with a plurality of stored characters representing known HTML tags. Once the type of HTML tag has been identified, a determination is made at step 450 as to whether or not the HTML tag is capable of influencing the format of the any subsequently displayed text. A format table of HTML tags capable of influencing the format of displayed text is stored within the memory of the computer. Referring to figure 5, the format table comprises a plurality of HTML tags which influence the format of displayed text together with a plurality of format values representing the style, font and font size associated with respective HTML tags. The characters representing the HTML tag are compared with each of the HTML tags stored within the format table until a match is located.

Once a match is located, the current format parameters are pushed onto the format stack and the current format parameters are set to equal those determined from the format table as corresponding to the matching HTML tag at step 455. Having determined the appropriate format for any subsequently displayed text, control is passed back to step 405. Any subsequently rendered text will be output according to the determined format.

If the determination at step 450 is negative, it is established at step 460 whether or not the tag corresponds to an applet tag. If so, an applet context object is created at step 465. Processing is then continued from step 405.

If the HTML tag does not represent an applet, it is established, at step 470, whether or not the tag represents a param tag. If so, the parameter data for the applet is stored within the most recently created or an appropriate applet context object at step 475 in preparation for the invocation of the applet. If the tag does not represent a param tag, processing of the HTML document is again continued from step 405 (nb it is standard for HTML browsers to simply ignore any tags which they do not recognise).

If the determination made at step 440 is such that the next non-space character is a solidus, a further determination is made at step 480 as to whether or not the characters stored within the text buffer represent an applet end tag, that is to say, whether or not the characters stored within the text stack are "</applet>". If so, an applet object is created at step 485 using the parameter data previously stored in applet context object and the applet is invoked, at step 490, using the init() Java instruction as is well known within the art to instigate execution of the applet. Processing continues then from step 405.

If the determination is such that the characters stored within the text buffer do not represent an end applet tag, it is established, at step 495, whether or not those characters represent an end tag associated with a tag capable of influencing the formatting of the text. If so, the format parameters most recently placed onto the format stack are then pulled or popped from the stack thereby discarding the current format parameters and setting the current format parameters to those popped from the stack. Therefore, any subsequently rendered text will be displayed using the current format parameters. Processing then continues from step 405.

As described above, the format stack stores a plurality of formats determined by from the various HTML tags which are capable of influencing the format of any subsequently output text. Each time a new HTML tag is encountered which is capable of influencing the format of subsequently output text, the current format parameters are pushed onto the stack and the current format parameters are set to equal those parameters which are associated with the new HTML tag. Therefore, any subsequently output text will have a format governed by the new format parameters. When the end tag associated with the above new HTML tag is encountered the current

format parameters are discarded and the format parameters most recently placed upon the stack are retrieved. Therefore, any subsequently output text is formatted according to the format parameters retrieved from the stack.

For example, a level one heading may have nested therein italicized text as follows: <H1>Level one Heading<I> Italic text </I>Return to level one</H1>. Due to the operation of the applet parser, the format parameters associated with level one heading tag, <H1>, would be retrieved and the text "Level one Heading" would be output according to those parameters. However, when the applet encounters the italic tag, <I>, the current format parameters, that is, the level one heading format parameters, are pushed onto the stack and the italic style is added to the current format parameters. Therefore, the subsequently rendered "Italic text" will be formatted according to the level one heading, but in italic style. When the end italic tag, </I>, is encountered, the current format parameters are discarded and the parameters most recently placed on the stack are retrieved. Therefore, the text "Return to level one" will be rendered according to the format parameters associated with the original level one heading tag.

Referring again to figure 5, it can be seen that the italic tag has associated therewith a font style of font.ITALICS while the remaining font types are represented by asterisks thereby indicating that the font name and size are wildcards. The effect of having such asterisks within a set of format parameters is that the values of the previous format parameter are retained for subsequent rendering of text. Therefore, the font name and font size of any subsequently rendered text remains the same as those for the previously rendered text. For example, assume that the <H1> tag has associated therewith the following set of format parameters, font.BOLD, 20, and TimesRoman and that the <I> tag has associated therewith format parameters of font.ITALIC, \*, and \*. Further, assume that the following HTML is to be rendered to the screen of the computer system: <H1>Text having <I> some italicised</I> text</H1>. The result of rendering the above text would be as follows:

#### Text having *some italicised text*

In the above flow chart the processing of repeated space characters is not expressly described. However, one skilled in the art would readily recognise that all but one of a plurality of contiguous space characters are ignored when rendering character or graphics to the screen.

It will be appreciated that the steps of rendering a first HTML document may be conducted substantially concurrently with the rendering of the second or an embedded HTML document. The respective documents may be processed using, for example, different threads within a multi-threaded operating system or environment for the execution of the applet performing the ren-

dering and the parsing. However, an embodiment can be realised in which all of the instructions contained in a preceding HTML document are parsed and rendered before the parsing and rendering of a succeeding or embedded HTML document is commenced. In such an embodiment the method steps of the present invention may be performed substantially sequentially.

## Claims

1. A method for rendering first and second html documents comprising respective first and second sets of html data to a display screen, wherein the first html document has embedded therein the second html document, comprising the steps of:

parsing and rendering the first set of html data to a first area of said screen,

reserving a second area of said screen for use in rendering said second set of html data, said second area being embedded within said first area, and

parsing and rendering the second set of html data to said screen within said second area.

2. A method as claimed in claim 1, wherein said first html document includes a Java applet tag containing a reference to a Java html parsing and rendering applet for parsing and rendering html to the screen, and wherein the step of reserving the second area is responsive to said Java applet tag, and said steps of parsing and rendering said second set of html data are performed by said Java applet.

3. A method as claimed in either of claims 1 or 2, further comprising the steps of creating a data structure for storing format data comprising a plurality of characteristics which determine the format of the rendering, said rendering being performed according to current format data, and wherein said steps of parsing and rendering further comprise the steps of:

identifying new format data,

storing a copy of said current format data in the data structure, and

modifying the current format data according to the new format data.

4. A method as claimed in claim 3, further comprising the steps of

identifying a need to change the current format

5

data,

recovering a copy of format data stored in the data structure, and

modifying the current format data according to the recovered format data.

10

5. A method as claimed in either of claims 3 or 4, wherein the step of modifying comprises the steps of changing only selectable ones of said plurality of characteristics of the current format data according to the new format data thereby retaining selectable characteristics of stored format data.

15

6. A system for rendering first and second html documents comprising respective first and second sets of html data to a display screen, wherein the first html document has embedded therein the second html document, comprising

20

means for parsing and rendering the first set of html data to a first area of said screen,

25

means for reserving a second area of said screen for use in rendering said second set of html data, said second area being embedded within said first area, and

30

means for parsing and rendering the second set of html data to said screen within said second area.

35

7. A system as claimed in claim 6, further comprising means for executing Java instructions, wherein said first html document includes a Java applet tag containing a reference to a Java html parsing and rendering applet for parsing and rendering html to the screen, and wherein said means for reserving the second area is responsive to said Java applet tag, and said means for parsing and rendering said second set of html data comprises said Java applet.

40

8. A system as claimed in either of claims 6 or 7, further comprising means for creating a data structure for storing format data comprising a plurality of characteristics which determine the format of the rendering, and wherein said means for rendering is responsive to current format data, and wherein said means for parsing and rendering further comprise

45

means for identifying new format data,

50

means for storing a copy of said current format data in the data structure, and

55

means for modifying the current format data according to the new format data.

9. A system as claimed in claim 8, further comprising:

means for identifying a need to change the current format data,

5

means for recovering a copy of format data stored in the data structure, and

means for modifying the current format data according to the recovered format data.

10

10. A system as claimed in either of claims 8 or 9, wherein the means for modifying comprises means for changing only selectable ones of said plurality of characteristics of the current format data according to the new format data thereby retaining selectable characteristics of stored format data.

15

11. A computer program product on a computer readable medium comprising a multi-levelled document object comprising a first level of html data, and a second level of html data embedded within or accessible from said first level, said first level of html data including a reference to said second level of html data to allow said second level of html data to be rendered within a predeterminable area of said first level of html data.

20

25

30

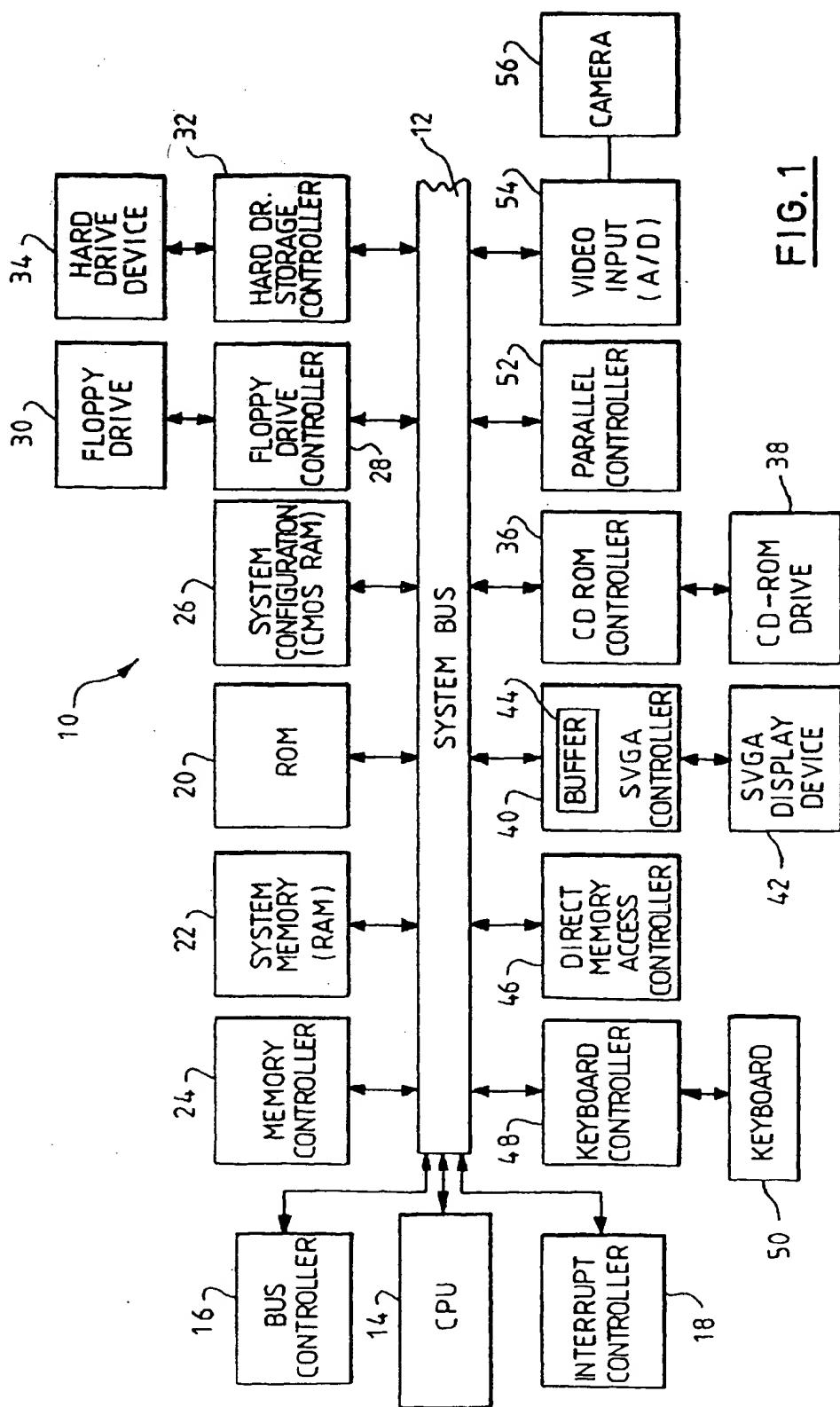
35

40

45

50

55

**FIG. 1**

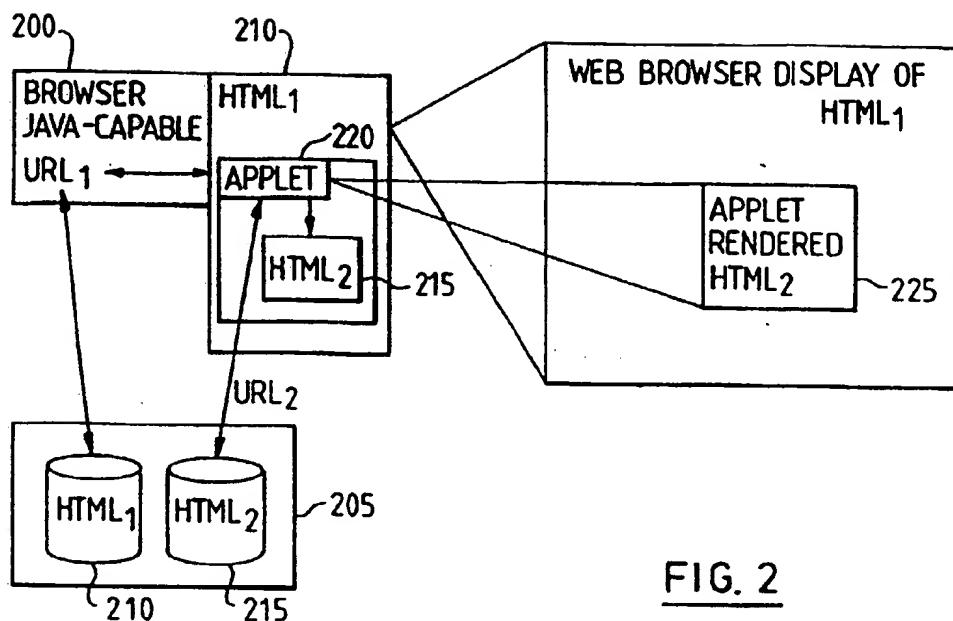


FIG. 2

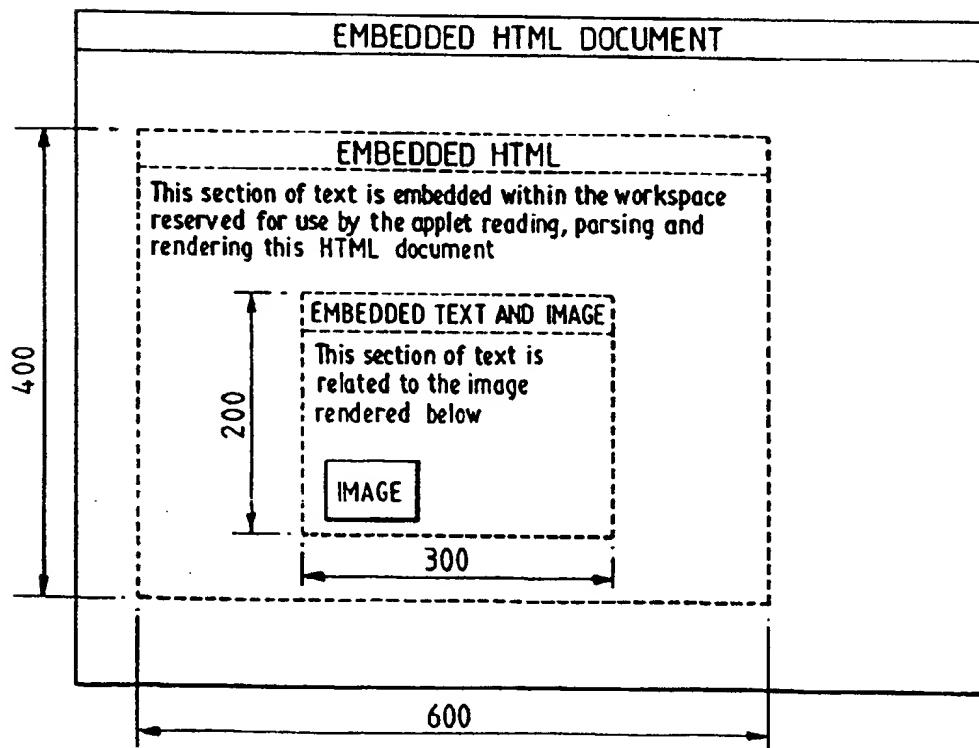


FIG. 3

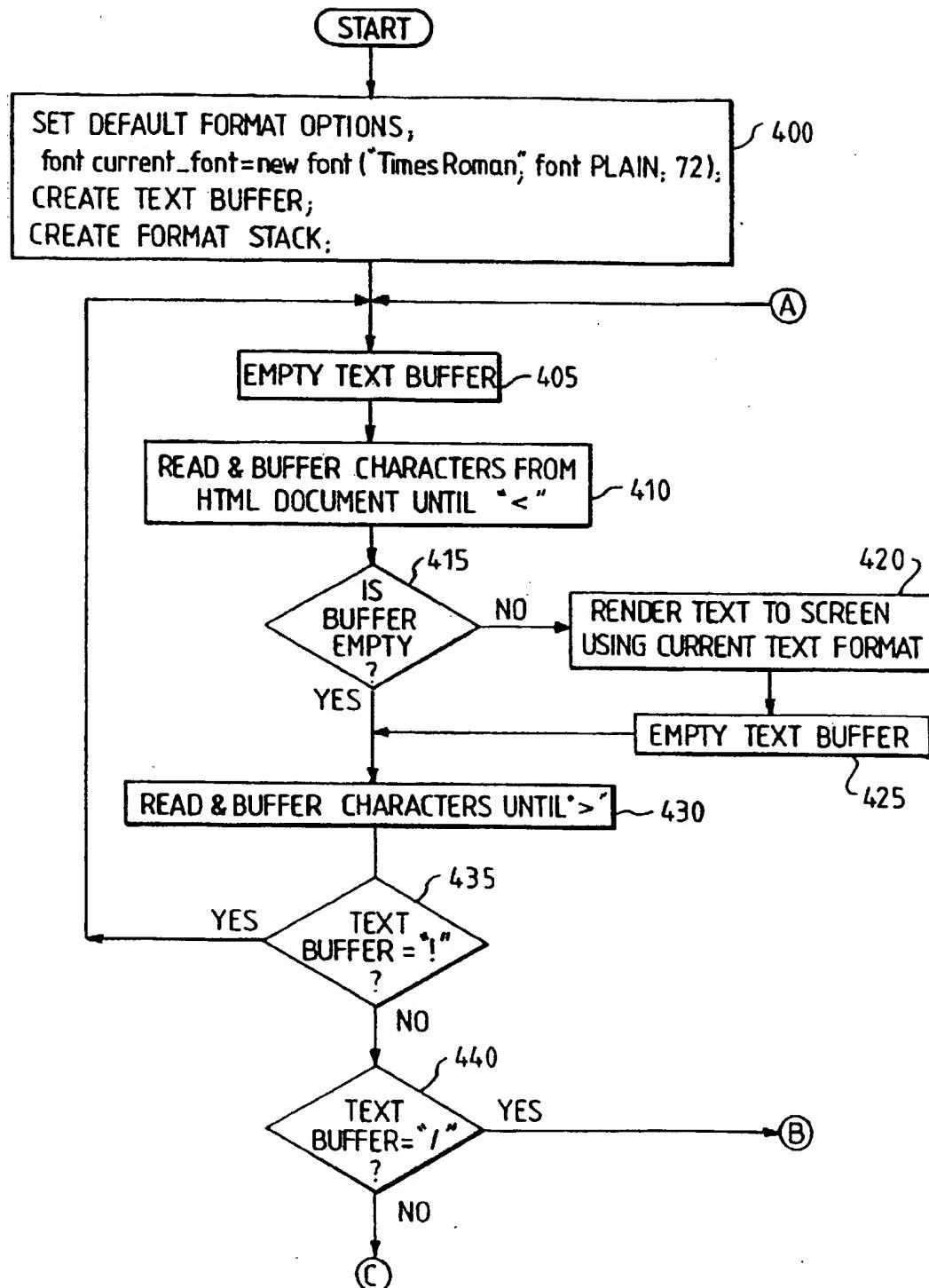


FIG. 4A

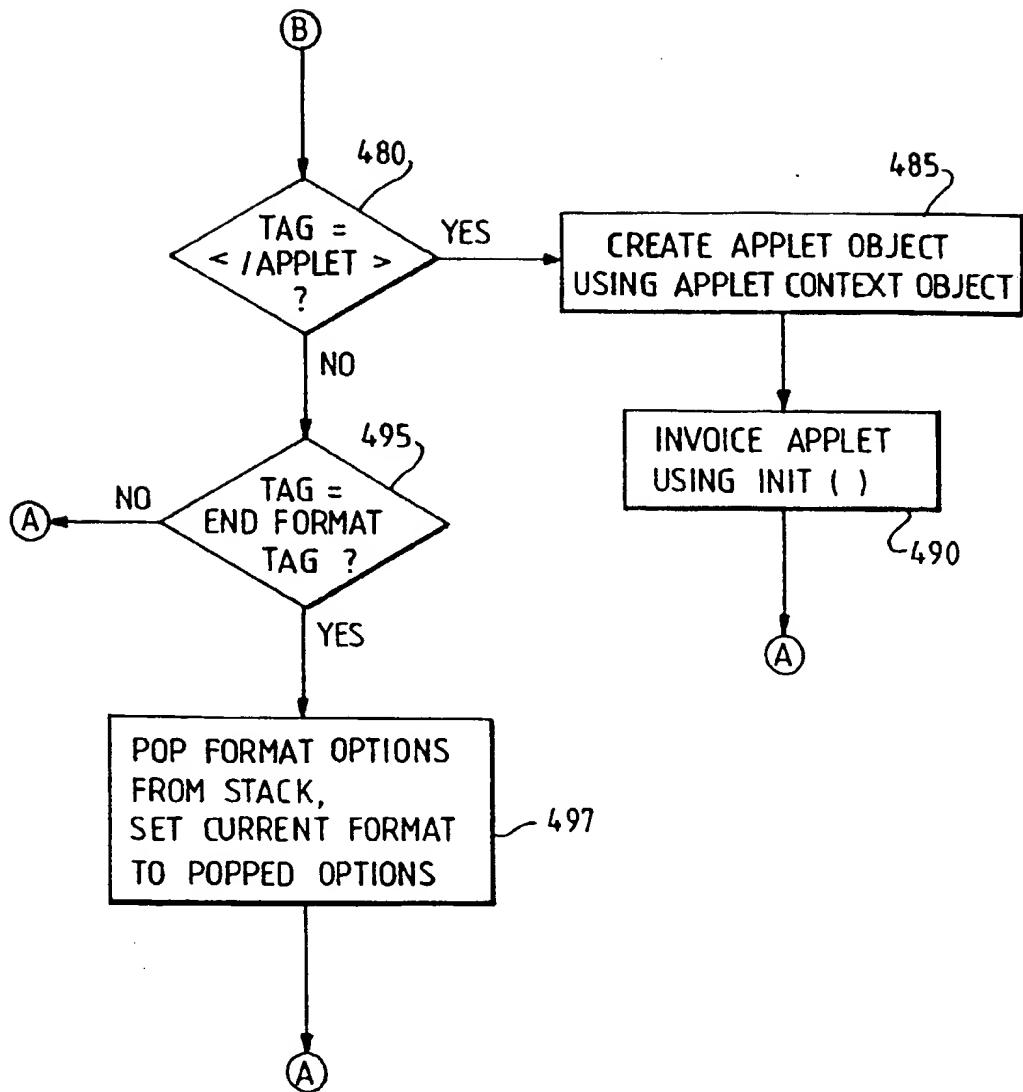


FIG. 4B

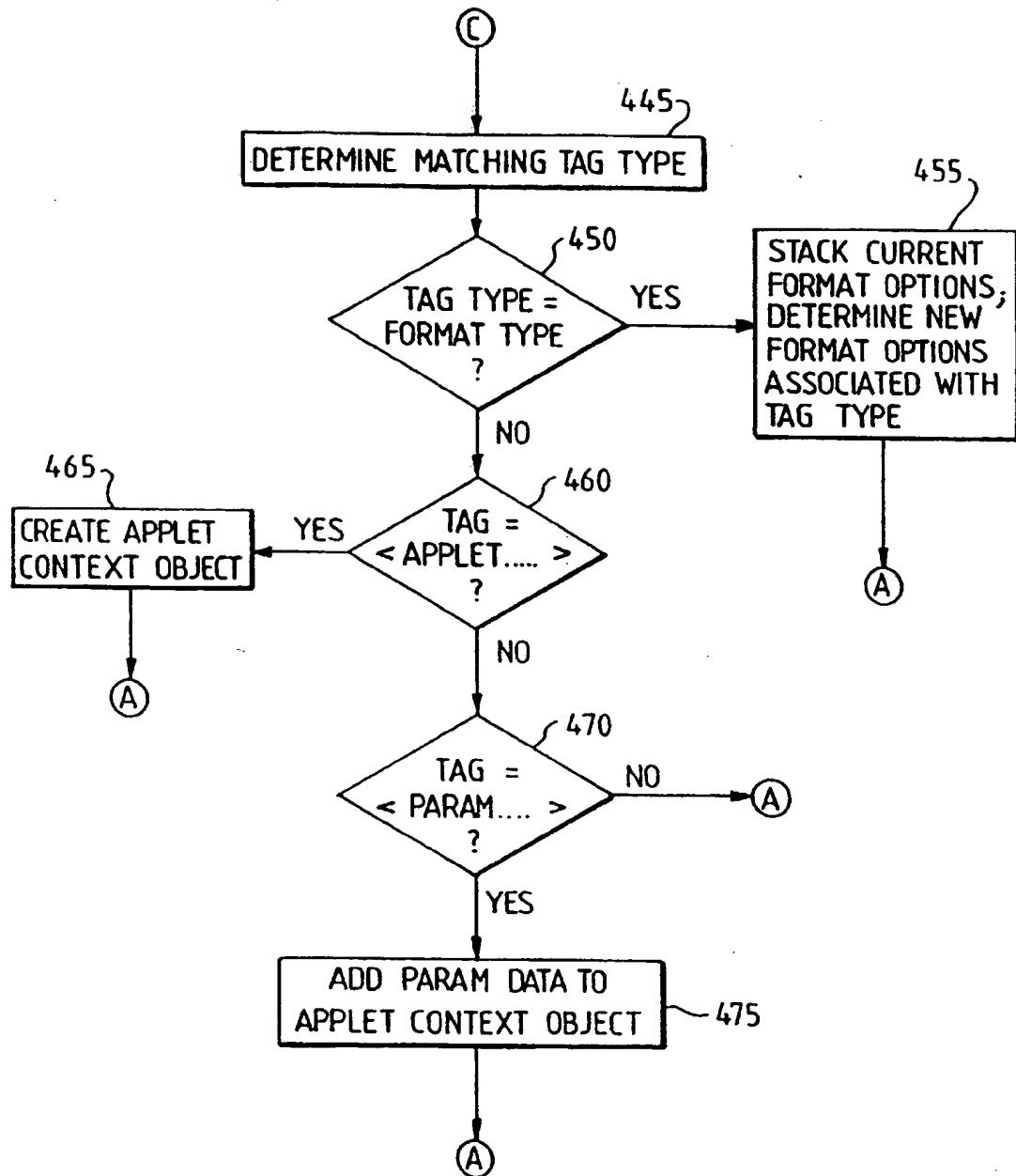


FIG. 4C

TAG	STYLE	FONT SIZE	FONT
< H 1 >	BOLD	20	TIMES ROMAN
< H 2 >	PLAIN	10	TIMES ROMAN
< I >	ITALIC	*	*
•	•	•	•
•	•	•	•
•	•	•	•

FIG. 5



European Patent  
Office

## EUROPEAN SEARCH REPORT

Application Number  
EP 97 30 4742

DOCUMENTS CONSIDERED TO BE RELEVANT			CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	
Y	BOUVIER D J: "THE STATE OF HTML" SIGICE BULLETIN, vol. 21, no. 2, 1 October 1995, pages 8-13, XP000585172 * page 8, left-hand column, line 1 - page 11, right-hand column, line 30; figures 1,2; tables 1-7 *	1,2,6,7, 11	G06F17/30 G06F17/22
Y	SCHLICHTER J H ET AL: "FOLIOPUB: A PUBLICATION MANAGEMENT SYSTEM" COMPUTER, vol. 21, no. 1, January 1988, pages 61-69, XP000052069 * page 64, column 2, line 26 - page 65, column 1, line 57; figures 2,3 *	1,2,6,7, 11	
A	EP 0 467 591 A (AMERICAN TELEPHONE & TELEGRAPH) 22 January 1992 * column 5, line 52 - column 7, line 36; figures 3,5 *	1,6,11	
A	"POSITIONING OF NESTED OR INCLUDED OBJECT PRESENTATION SPACES AND OBJECT DATA" IBM TECHNICAL DISCLOSURE BULLETIN, vol. 32, no. 9B, 1 February 1990, pages 320-323, XP000082366 * the whole document *	1,6,11	TECHNICAL FIELDS SEARCHED (Int.Cl.6) G06F
A	"SCATTERING AND GATHERING DATA AMONG PRESENTATION SPACES" IBM TECHNICAL DISCLOSURE BULLETIN, vol. 32, no. 10B, 1 March 1990, pages 24-27, XP000097788 * the whole document *	1,6,11	
		-/-	
The present search report has been drawn up for all claims			
Place of search EPO FORM 1503/02 (P04/C01)	Date of completion of the search 16 October 1997	Examiner Deane, E	
CATEGORY OF CITED DOCUMENTS		T : theory or principle underlying the invention E : earlier patent document, but published on, or after the filing date D : document cited in the application L : document cited for other reasons  A : member of the same patent family, corresponding document	
X : particularly relevant if taken alone Y : particularly relevant if combined with another document of the same category A : technological background O : non-written disclosure P : intermediate document			



European Patent  
Office

## EUROPEAN SEARCH REPORT

Application Number  
EP 97 30 4742

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
A	GROSS C: "LES APPLICATIONS JAVA VOYAGENT PAR RESEAU" ELECTRONIQUE, no. 56, 1 February 1996, page 8/9 XP000554834 * the whole document * ----	2,7	
P,X	PATENT ABSTRACTS OF JAPAN vol. 097, no. 006, 30 June 1997 & JP 09 044383 A (RICOH CO LTD), 14 February 1997, * abstract * -----	1,6,11	
			TECHNICAL FIELDS SEARCHED (Int.Cl.5)
<p>The present search report has been drawn up for all claims</p>			
Place of search	Date of completion of the search	Examiner	
BERLIN	16 October 1997	Deane, E	
CATEGORY OF CITED DOCUMENTS		T : theory or principle underlying the invention E : earlier patent document, but published on, or after the filing date D : document cited in the application L : document cited for other reasons & : member of the same patent family, corresponding document	
X : particularly relevant if taken alone Y : particularly relevant if combined with another document of the same category A : technological background O : non-written disclosure P : intermediate document			